AFRL-RH-WP-TR-2017-0045

Verifiable Task Assignment and Scheduling Controller

Clayton Rothwell
Infoscitex Corporation
Dayton, OH 45431

Michael Patzek
Airman Systems Directorate
Wright-Patterson AFB, OH 45433

Laura Humphrey
Aerospace Systems Directorate
Wright-Patterson AFB, OH 45433

July 2017

Interim Report

AIR FORCE RESEARCH LABORATORY
711 HUMAN PERFORMANCE WING,
AIRMAN SYSTEMS DIRECTORATE,
WRIGHT-PATTERSON AIR FORCE BASE, OH 45433
AIR FORCE MATERIEL COMMAND
UNITED STATES AIR FORCE

NOTICE AND SIGNATURE PAGE

AFRL-RH-WP-TR-2017-0045 HAS BEEN REVIEWED AND IS APPROVED FOR PUBLICATION IN ACCORDANCE WITH ASSIGNED DISTRIBUTION STATEMENT.

//signed//
GLORIA CALHOUN
Work Unit Manager
Supervisory Control and Cognition Branch

//signed//
JOSEPH C. PRICE, MAJ, USAF
Acting Chief, Supervisory Control Cognition Branch
Warfighter Interface Division

//signed//
KRISTOFFER A. SMITH RODRIGUEZ, LTCOL, USAF
Acting Chief, Warfighter Interface Division
Airman Systems Directorate
711 Human Performance Wing

REPORT DOCUMENTATION PAGE	*Form Approved* *OMB No. 0704-0188*

1. REPORT DATE *(DD-MM-YYYY)*	**2. REPORT TYPE**	**3. DATES COVERED** *(From - To)*
09-06-2017	Interim	April 2012 – Dec 2016

4. TITLE AND SUBTITLE Verifiable Task Assignment and Scheduling Controller	**5a. CONTRACT NUMBER** FA8650-14-D-6500/0002
	5b. GRANT NUMBER
	5c. PROGRAM ELEMENT NUMBER
6. AUTHOR(S) Clayton Rothwell* (Infoscitex Corp.) Michael Patzek and Laura Humphrey (Air Force Research Laboratory)	**5d. PROJECT NUMBER**
	5e. TASK NUMBER
	5f. WORK UNIT NUMBER H0JC (53290904)
7. PERFORMING ORGANIZATION NAME(S) AND ADDRESS(ES) 711 HPW/RHCI 2210 Eight Street Wright-Patterson AFB, OH 45433	**8. PERFORMING ORGANIZATION REPORT NUMBER**
9. SPONSORING / MONITORING AGENCY NAME(S) AND ADDRESS(ES) Air Force Materiel Command *Infoscitex Corp. Air Force Research Laboratory 4027 Colonel Glenn Hwy 711 Human Performance Wing Beavercreek OH 45431 Airman Systems Directorate Warfighter Interface Division Supervisory Control and Cognition Branch Wright-Patterson AFB OH 45433	**10. SPONSOR/MONITOR'S ACRONYM(S)** 711 HPW/RHCI
	11. SPONSOR/MONITOR'S REPORT NUMBER(S) AFRL-RH-WP-TR-2017-0045

12. DISTRIBUTION / AVAILABILITY STATEMENT

DISTRIBUTION STATEMENT A. Approved for public release: distribution unlimited.

13. SUPPLEMENTARY NOTES

88ABW Cleared 08/23/2017; 88ABW-2017-4079.

14. ABSTRACT

Anticipated advances in the use and capability of unmanned systems will result in increased demands on mission planning and human-machine teaming. An approach to handle these demands is through a model checking tool for verification of unmanned aerial vehicle mission planning. This report describes the development and testing of one such tool, Specification Pattern Editor and Checker (SPEC), with an emphasis on providing the human operator a way to communicate their high-level mission goals and objectives to the model checking software without having to learn temporal logics. This initial verification tool was evaluated for usability and further refined into a second version. The tool was experimentally tested to investigate the effects of verification tools like model checkers for pre-mission route planning. Results showed that a verification tool increased mission planning accuracy, while also increasing mission planning time compared to mission planning with a baseline configuration without verification. Subjective workload was not different between the two display configurations. These results indicate that verification tools such as SPEC enable humans to communicate mission objectives and constraints to machines, which results in more accurate plans, and that further research on speeding up the interaction could result in further gains from verification tools.

15. SUBJECT TERMS Unmanned Aerial Vehicle/Unmanned Systems Mission Planning, Human Autonomy Interface, Human Autonomy Teaming, Verification and Validation, Model Checking, Mission Specification

16. SECURITY CLASSIFICATION OF:			**17. LIMITATION OF ABSTRACT**	**18. NUMBER OF PAGES**	**19a. NAME OF RESPONSIBLE PERSON** Gloria Calhoun
a. REPORT	**b. ABSTRACT**	**c. THIS PAGE**			**19b. TELEPHONE NUMBER** *(include area code)*
Unclassified	Unclassified	Unclassified	SAR	59	

Standard Form 298 (Rev. 8-98)
Prescribed by ANSI Std. 239.18

THIS PAGE IS INTENTIONALLY LEFT BLANK

TABLE OF CONTENTS

LIST OF FIGURES

LIST OF TABLES

EXECUTIVE SUMMARY

Anticipated advances in the use and capability of unmanned systems will result in increased demands on mission planning and human-machine teaming. An approach to handle these demands is through a model checking tool for verification of unmanned aerial vehicle mission planning. This report describes the Specification Pattern Editor and Checker (SPEC), a usability test of the tool, and an experiment to test the benefits of mission verification. SPEC supports communication between an operator, a UAV ground control station, and model checking software by providing a set of tools to write specifications in English that can be converted to temporal logic. The long-term goal for the envisioned capability would be for humans to communicate specifications and for automated routines to generate satisfactory mission plans for all the UAVs via synthesis. However, as a preliminary step, our short-term goal here is to equip an operator with a tool that can check human-generated plans and help achieve the mission objectives while complying with the rules of engagement and constraints of the mission.

The tool was experimentally tested to investigate the effects of verification tools like model checkers for pre-mission route planning. Results showed that a verification tool increased mission planning accuracy, while also increasing mission planning time compared to mission planning with a baseline configuration without verification. Subjective workload was not different between baseline configuration and the verification tool configuration. These results indicate that verification tools such as SPEC enable humans to communicate mission objectives and constraints to machines, which results in more accurate plans, and that further research on speeding up the interaction could result in further gains from verification tools.

1.0 INTRODUCTION

Over the last two decades, military operations with unmanned aerial vehicles (UAVs) have become increasingly complex and dynamic. Missions rely on tightly coupled coordination between vehicles and teammates in the air and on the ground. Moreover, development efforts are underway to give a single operator simultaneous control of multiple UAVs, which could entail operating in extremely high tempo and high cognitive workload situations. Future systems may add further complexity through decentralized control elements and flexible autonomy, such as autonomy that can be operator-customized for the mission at hand. One concept for supporting operator decision-making under these demands is through a mission planning and management verification tool.

Recently, researchers have applied formal methods – mathematically-based languages, techniques, and tools used to design and verify the safe and reliable operation of systems – to robotic systems, including UAVs (Humphrey, 2012; Doherty, Kvarnstrom & Heintz, 2009; Humphrey & Patzek, 2013). Formal methods include model checking, a technique in which a state-based representation of a system is automatically checked against a set of desired specifications generally expressed in temporal logic. In the UAV domain, model checking can be used to verify that UAV assignments, routes, and on-board dynamic mission planning conform with mission specifications (e.g., objectives, constraints, and rules of engagement) during mission planning and during mission execution, thereby increasing the chances of mission success. There are also approaches for automatically synthesizing (i.e., generating) mission plans that conform to the specifications (Kress-Gazit, Fainekos & Pappas, 2008; Humphrey, Wolff & Topcu, 2014).

Formal methods could be used to provide a better framework for communication between human and autonomous system members, a vital element for successful collaboration/teaming as identified by researchers in human-automation interaction (Klein, Woods, Bradshaw, Hoffman & Feltovich, 2004). The mission specifications would communicate the human's intent to the system, which in the case of synthesis, would create mission plans to accomplish that intent, or in the case of verification, would check that the human-created plans accomplish the intent. However, formal methods and

the associated temporal logics can be difficult for a novice to understand, use, and learn (Dwyer, Avrunin & Corbett, 1999). Therefore, in order to use formal methods to communicate the operator's specifications to autonomous UAVs, an interface is required that captures an operator's intended meaning and expresses it in a formal methods framework.

This report will describe the Specification Pattern Editor and Checker (SPEC), a usability test of the tool, and an experiment to test the benefits of mission verification. SPEC supports communication between an operator, a UAV ground control station, and model checking software by providing a set of tools to write specifications in English that can be converted to temporal logic. The long-term goal for the envisioned capability, one that is more challenging and perhaps more rewarding, would be for humans to communicate specifications and for automated routines to generate satisfactory mission plans for all the UAVs via synthesis. However, as a preliminary step, our short-term goal here is to equip an operator with a tool that can check human-generated plans and help achieve the mission objectives while complying with the rules of engagement and constraints of the mission. With model checking tools integrated into the control station, it is anticipated that operators will perform complex mission planning and management tasks for single- and multi-UAV operations more effectively.

There is a long and varied list of potential mission specifications within the multi-UAV domain because of the widespread use of UAVs in many types of missions. In order to focus this initial interface work, a convoy escort mission was selected. In this mission, the primary goal is to assist a convoy of ground vehicles in quickly and safely reaching its destination by providing overhead surveillance: looking ahead along the route, scouting alternate routes, investigating traffic congestion and suspicious situations, and periodically checking possible bottlenecks. To establish detailed system requirements, in-house personnel with experience evaluating multi-UAV surveillance technology participated in a cognitive task analysis (CTA). For the CTA, each participant stepped through a convoy escort scenario using a "think aloud" protocol in which the participants explained their thought processes, strategies, and decisions to an experimenter/recorder. The scenario was designed to be complex and demanding. It incorporated UAVs with heterogeneous capabilities, dynamic events that rerouted the

convoy, and instances in which the vehicles available to the operator could not fulfill all the task demands of the situation (more details are reported in Stanard, Bearden, & Rothwell, 2013). The CTA results informed the present work, indicating how operators plan and conduct these types of missions, how they would direct multiple semi-autonomous UAVs to meet mission specifications, and how they would attempt to characterize the mission specifications.

The next two sections start by describing the SPEC interface and a usability study of SPEC. The two sections after that describe changes made to SPEC based on the usability study and a follow-up experiment. The last section concludes with a summary of SPEC and the study and experiment results.

2.0 SPEC INTERFACE

The SPEC components and interface (i.e., the SPEC tool) described below are integrated within the Verifiable Task Assignment and Scheduling Controller (VTASC) system (Figure 1). The VTASC system has many components: a human operator, UAVs that act in a semi-autonomous fashion, a UAV ground control station known as Vigilant Spirit, a piece of software called the model builder, and the NuSMV model checker.

The human operator performs supervisory control of multiple UAVs, monitors their mission activities, views the images they collect, and has the ability to direct them through autopilot commands and waypoint-based flight plans. Currently, the UAVs receive and execute autopilot commands. However, we expect that future UAVs will have a flexible control scheme so they can operate across a spectrum of varying levels of automation, ranging from teleoperated to fully autonomous. On the autonomous side of the spectrum, UAVs would behave as intelligent systems, detecting dynamic elements of environments and missions and changing their behavior in response. Therefore, our system has been developed to accommodate these envisioned autonomous UAV capabilities in future tests.

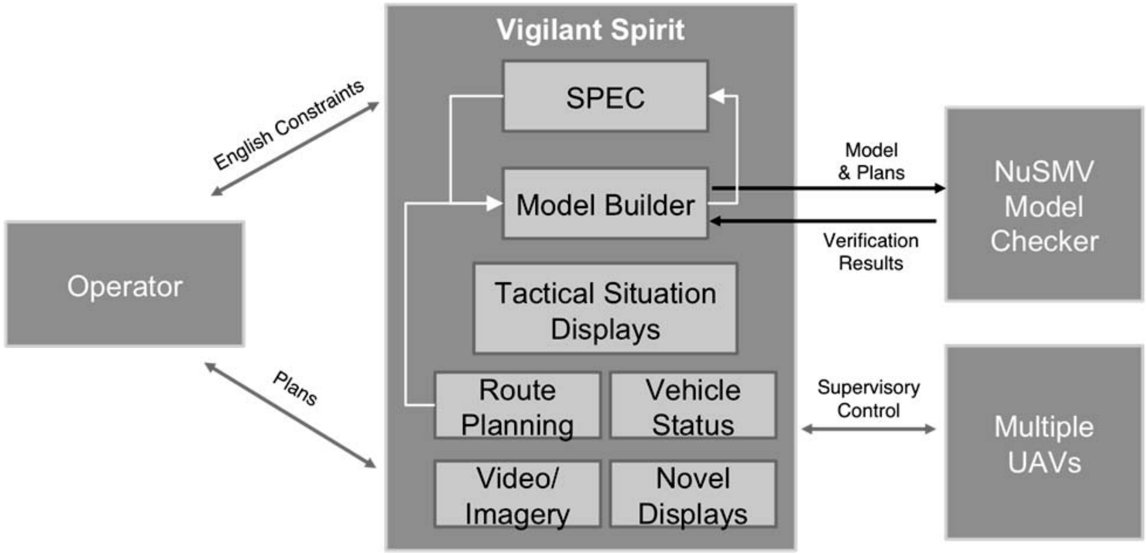


Figure 1. Diagram of VTASC system components that displays the connections between the operator, the Vigilant Spirit Control Station, the NuSMV model checker, and UAVs

Our system builds off the Vigilant Spirit Control Station (VSCS), a test bed designed by the Air Force Research Laboratory for studying multi-UAV supervisory control and associated interfaces and technologies (Rowe, Liggett, & Davis, 2009; Feitshans, Rowe, Davis, Holland, & Berger, 2008). In addition to SPEC, VSCS has tactical situation displays (i.e., geo-spatial maps), vehicle status displays, route planning interfaces for creating vehicle flight plans, and video/imagery displays (Figure 2). The model builder software is a unit within VSCS that constructs a mission model that is sent to the model checker, and it interfaces with the model-checking software (e.g., initiates the model checker, receives the results of the check). The model created by the model builder contains representations of the vehicles, their capabilities, and their flight plans as well as the environment. The specifications output by the SPEC interface are passed to the model builder, which adds them to the model and sends the model to the New Symbolic Model Verifier (NuSMV) model checker. NuSMV is a university-developed symbolic model checker that accommodates many temporal logics (Cavada et al., 2010; Cimatti et al., 2002) including linear temporal logic (LTL), computation tree logic (CTL), and real-time computation tree logic (RTCTL).

VSCS has user interfaces for UAV control/mission management and the SPEC tool (Figure 2). VSCS has a configurable layout for different missions and different tools. For VTASC, VSCS has a two-monitor layout with the tactical situation display (TSD), vehicle

status, and route planning on the left screen and the SPEC tool on the right screen. No video/imagery displays are present in this layout because in pre-mission planning there is no live video to inspect.

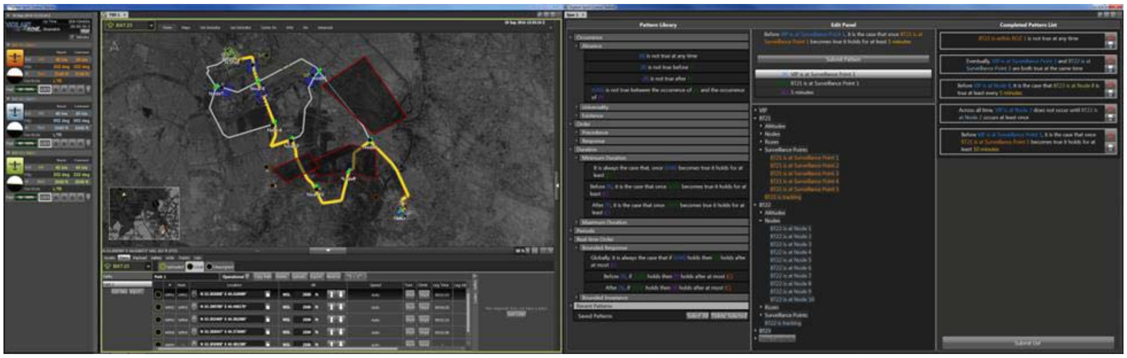


Figure 2. Screen shot of the Vigilant Spirit Control Station with a two monitor configuration. On the left is the TSD, vehicle status display, and route planning interfaces. On the right is the SPEC tool

As noted earlier, the model contains specifications and vehicle plans. Both are created by the operator and added to the model. The long-term vision for this system is to have an automatic planning tool that uses the specifications to synthesize satisfactory vehicle plans so that the operator can communicate desirable system behavior to the UAV's on-board controller in natural language and have the UAV automatically carry out planning and re-planning as the mission unfolds. Automatic planning has not been implemented yet in VTASC. Consequently, the tool currently is used to verify vehicle plans created by the operator. To do this, the operators use the existing VSCS route planning interfaces to create flight plans for each UAV. A custom software method within the model builder converts the flight plans into the format needed by the model checker (i.e., a finite state transition system). Flight plans consist of legs of travel that are defined by waypoints. Each leg has a speed and altitude associated with it, and each waypoint can have loiter operations associated with it. The model builder uses a simple air vehicle model that assumes instantaneous turning and linear climbing/descending to represent vehicle plans as finite-state transition systems. Conceptually, the model checker verifies whether or not the vehicle plans the operator created will accomplish what the operator had intended (as conveyed by the specifications). The following section will detail how operators input specifications into the SPEC tool.

2.1 Specification Patterns

The SPEC interface uses a method for developing specifications that is based on temporal logic patterns rather than a grammar-based translation approach (such as the approached used by the LTLMoP tool, Kress-Gazit et al., 2008; and our past work, Rothwell et al., 2013). We note this pattern-based approach is also used by the Specification Pattern Instantiation and Derivation EnviRonment (SPIDER) tool suite (Konrad & Cheng, 2005a). However, SPEC has many additional interface functionalities and is tightly integrated within UAV management software.

In the pattern-based approach, a temporal logic pattern captures a temporal relationship that expresses a commonly desired system property. Temporal relationships, such as *property P always happens at least once every t minutes,* are generic in nature such that *P* can be any property of a given system and *t* can be any one of the discrete time steps of that system. Patterns can be instantiated with variables specific to a particular system of interest and then used in model checking. The next section will describe the features of SPEC for selecting a pattern, instantiating a pattern, and using the model checker to check the pattern against current vehicle flight plans.

2.2 SPEC Panels

The SPEC tool contains three panels that are organized for a left-right workflow. The three panels are: the Pattern Library, the Edit Panel, and the Completed Pattern List. Figure 3 shows an expanded view of the SPEC tool, shown previously in Figure 2. Descriptions of each panel and the interactions between panels are given below.

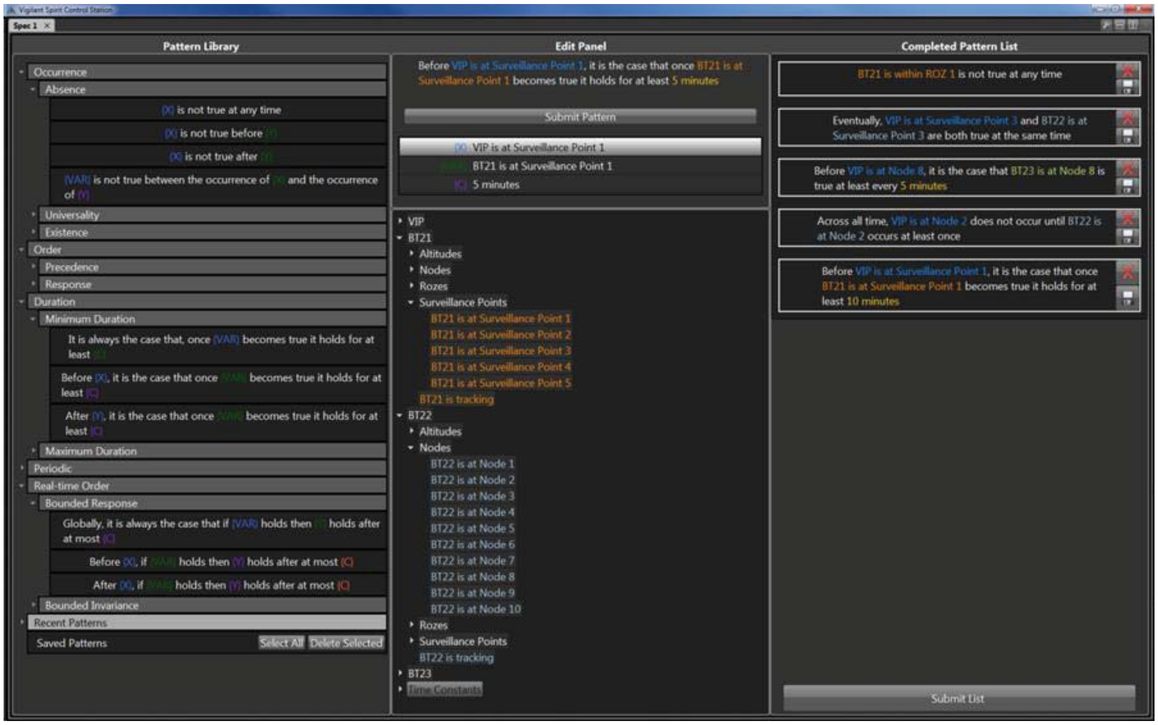


Figure 3. Expanded view of the SPEC tool showing the three panels (from left to right): Pattern Library, Edit Panel, and Completed Pattern List

Pattern Library

The pattern library contains all the patterns that can be used in an organized and interactive tree view. Patterns are categorized as: occurrence, order, periodic, duration, and real-time order. These categories came from two extensive surveys of specification literature and model checking applications (Dwyer et al., 1999; Konrad & Cheng, 2005b). Occurrence patterns are relationships that capture whether something should always occur, never occur, or occur at some point in the future. Order patterns are relationships that capture whether something should occur before or after something else. Periodic patterns are relationships that capture whether something should repeatedly occur within some time window. Duration patterns are relationships that capture whether something should occur for more or less than a certain length of time. Real-time order patterns are similar to order patterns, which use before and after, but add the ability to specify how much time before or after.

A list of patterns used in the first version of SPEC is contained in Appendix A. Most patterns come from the literature, but some patterns were customized for this domain (e.g., to create the comparisons: equal to, greater than, less than). The English language representation of each pattern was taken from Konrad and Cheng (2005b), then

adapted in an attempt to make it easier to read and understand. Our process for adapting the English representation of patterns was: be as consistent as possible in the use of variables and words, use "happens" instead of "is true" or "occurs," and simplify understanding by rewording patterns to avoid negatives and double negatives wherever possible.

Each category of patterns is collapsed by default and can be expanded to display the patterns in that category. Categories of patterns have a distinct look and feel from the patterns themselves, to make them clearly distinguishable. Once a category is expanded, either individual patterns are displayed and can be selected, or subcategories are displayed and can be expanded if there are a large number of patterns in the category. For example, in the Occurrence patterns, there are three sub-categories: Absence, Universality, and Existence. Upon selection of a pattern, the pattern is highlighted to provide visual feedback of the selection, and the pattern populates the pattern editor. Selecting a pattern from the library replaces anything that was previously in the pattern editor window.

The patterns listed in the library are pulled from a configuration file, which makes it easy to modify patterns or add new patterns. Patterns in this configuration file are written by an expert and must have both a temporal logic and English representation. The pattern library also includes two other categories that are dynamic: recent patterns and saved patterns. The recent patterns category includes the 10 most recently completed patterns. Saved patterns are stored in a file from session to session in order to give the user the option to save frequently used patterns from mission to mission.

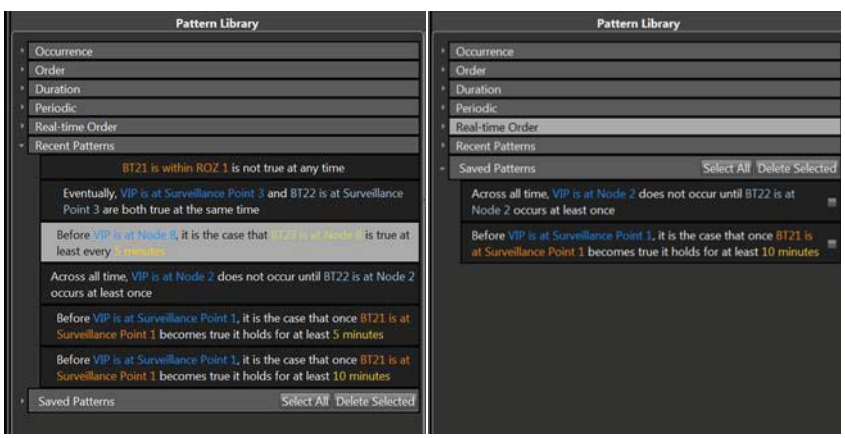


Figure 4. Recent patterns (left) and saved patterns (right) in the Pattern Library panel of SPEC

Edit Panel

In the top of the Edit Panel (see Figure 3), the selected pattern is displayed. Immediately beneath that is a list of the variables contained in the pattern that must be assigned a value in order to complete the specification. If a pattern has 2 variables (e.g., VAR and X), then each variable is displayed on a separate row. The first variable's row is selected by default, and the selected variable can be changed by clicking on another variable's row. Potential variable assignments are pulled from a representation of the mission map and vehicles used by the model builder. Assignments include the UAV's altitude, the UAV's location above a labeled road intersection, the UAV's location above possible convoy bottlenecks, and the UAV's location within no-fly-zones. Unlike the patterns, the variables in the model cannot be easily configured. Updates to the model require changes to the model builder software and possibly the air vehicle model. Future work could enable the addition of new variables by allowing this model to be more easily modified.

Once all variables have been assigned, the completed specification can be submitted to the completed pattern list. If the variables have not been assigned and the user attempts to submit the specification, an error message will appear telling the user to assign all the variables. If the specification is a duplicate of one that has already been submitted, an error message will appear.

Completed Pattern List

The completed pattern list (see Figure 3) holds the instantiated patterns, now considered mission specifications. Each completed pattern can be edited, saved, or deleted. At the bottom of the completed pattern list is a button to submit the list of specifications to the model checker. After clicking this button, the completed pattern list first shows that the model checker is checking the vehicle plans against the mission specifications, as indicated by the blue ellipsis symbol to the left of each pattern (Figure 5 left panel). Once the check is complete, the ellipsis symbols disappear and an icon of either a green checkmark symbol or red prohibition symbol indicates the result for each pattern (Figure 5 right panel). Each specification is checked independently in the NuSMV software. Independent checks allow the interface to provide individual feedback on each specification. With the method of individual checks, it is possible to submit contradictory

or conflicting specifications, e.g., UAV-1 never flies over Point 1 and UAV-1 eventually flies over Point 1. This example is a trivial case that is obviously contradictory, and though it is possible that non-trivial cases may exist, we did not implement a check for conflicting specifications. (For more discussion of specification conflicts that lead to unsatisfiable constraints, see Kim & Humphrey, 2014). Also, the specification checks are not run together but are broken into two groups: one for relative time specifications (e.g., LTL) and the other for real-time specifications (e.g, RTCTL). These checks are executed by separate instances of NuSMV that run in parallel. In some instances there is a delay between receiving the results of the two checks.

Figure 5. The Completed Pattern List during two states of operation. On the left, the model checker is computing a solution. On the right, final model checking results are displayed for each specification

3.0 USABILITY TEST

3.1 Goals

The goals of the test were to evaluate the usability of the SPEC tool within the context of a mission use case. Specifically, we solicited feedback on the pattern concept, the method of creating specifications with the SPEC tool, and presentation of the model checker output.

3.2 Procedure

The usability evaluation involved six participants. These individuals came from the population of scientists and engineers at the Air Force Research Laboratory, and none had prior experience with SPEC or VTASC. First, participants were introduced to the project and the goals of the test. Second, participants received training on VSCS, on how

to create mission plans in VSCS, and the SPEC tool. The training used demonstrations and hands-on exercises with at least one pattern from each category in the pattern library to ensure participants were familiarized with the technology prior to the usability test. Training length varied with prior knowledge of VSCS, but on average was 1.25 hours long. After training, the participants learned about the task they were going to perform as part of the usability test. The task employed was a mission planning scenario. The mission scenario was based on convoy escort in an urban environment using 3 UAVs. The task was terminated by either 1) correctly creating all 8 specifications and building mission plans that met the specifications or 2) after 30 minutes elapsed, whichever came first. After the task, participants filled out a usability questionnaire (described below). Once the participants finished the questionnaire, the test administrator conducted a debriefing session and semi-structured interview. The purpose of the debriefing session was to capture free response comments from participants as well as clarify any written comments on the questionnaire (e.g., illegible handwriting, ambiguous statements, unfamiliar terms). The contents of the semi-structured interview are described below. On average, the whole procedure took 2.25 hours to complete.

Mission Scenario

The mission scenario was in the domain of convoy escort and had a combination of spatial and temporal constraints. In this scenario, a convoy of vehicles containing a very important person (VIP) begins its route at Node 1 and ends at Node 8 (shown in Figure 6). The primary convoy route is shown as a black dashed line, with alternate routes shown in white. Critical segments of the route are the intersections between the primary and alternate routes and the bridges. The intersections are critical because they are decision points for continuing on the primary route or switching off the primary route to an alternate route. The bridges are critical because they are known points of vulnerability and high exposure. Bridges are called surveillance points (SPs) and have to be inspected from an altitude below 2400 ft MSL, due to the simulated sensor capabilities on these aircraft. The altitude space is banded in 200 ft bands, starting at 2000 ft and ending to 3599 ft and some UAVs have to be in separate bands during the whole mission. In addition, some current operations and known threats may create restricted operating zones (ROZs), or regions of airspace that are ‘no fly’ areas for these UAVs. The mission

objectives for the scenario are listed below. These objectives were selected because they were consistent with the result from the cognitive task analysis and a good cross-section of the different categories of patterns in SPEC.

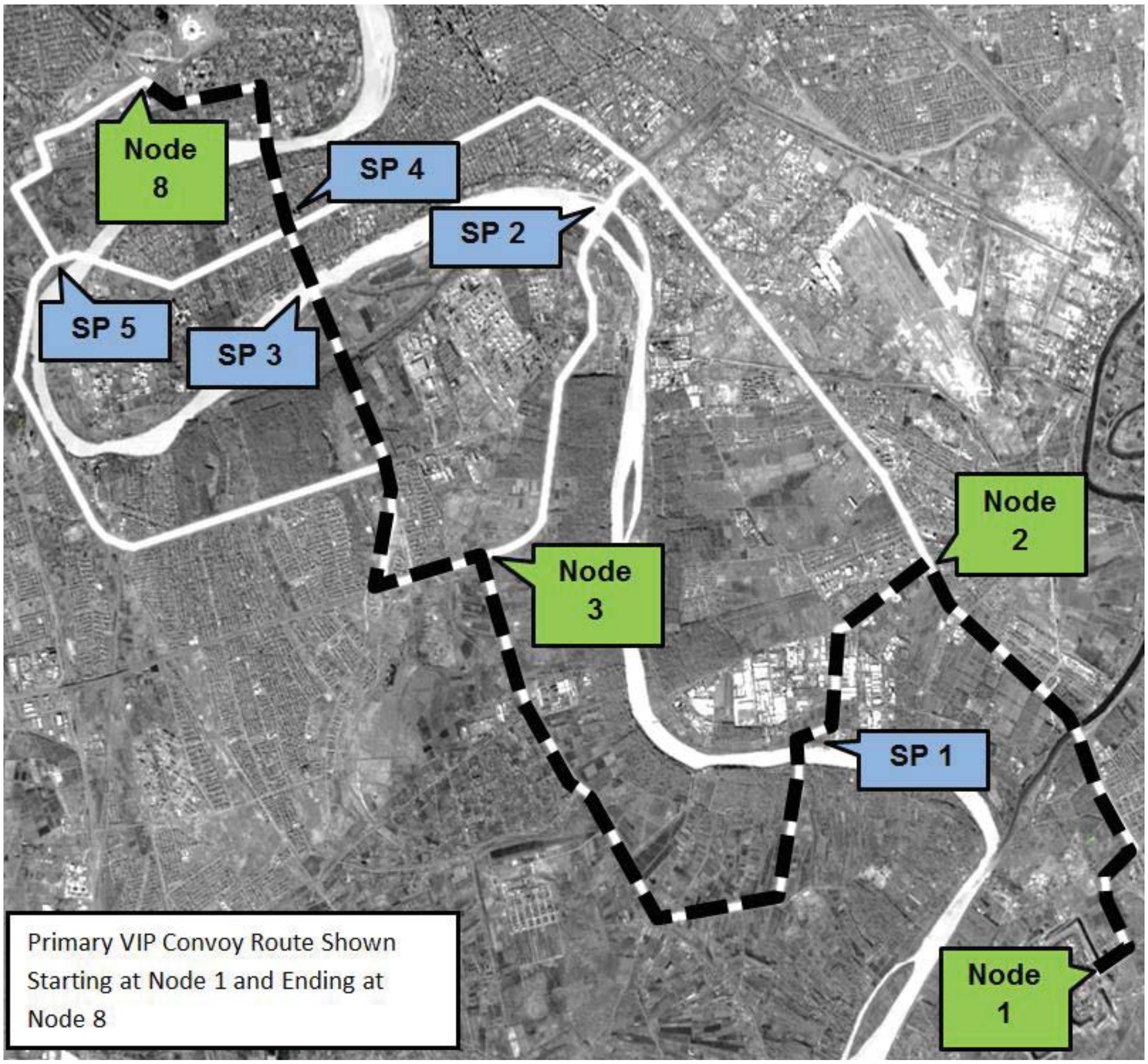


Figure 6. The usability test mission map. Node 1 is the point where the VIP convoy starts. Node 8 is the destination point for the VIP convoy. The primary route is identified as a black dashed line, and alternative routes are identified as white bold lines. Other important points are listed as other Nodes or Surveillance Points (SP), such as Node 2, the first decision point, or SP1, the first bridge.

Mission Objectives for the Usability test

1) A UAV arrives at the destination (Node 8) before the VIP
2) A UAV arrives at the first route decision point (Node 2) before the VIP
3) A UAV visits the destination to take a picture every 5 minutes until the VIP arrives
4) A UAV takes a picture of SP1 every 3 minutes until the VIP goes through SP1

5) A UAV takes a picture of SP3 every 3 minutes after the VIP goes through SP1
6) All UAVs avoid all Restricted Operating Zones
7) UAV 2's altitude should always be different than UAV 3's altitude
8) A UAV is located at SP2 to provide overwatch when VIP goes through

Questionnaire and Interview

The questionnaire was adapted from the Questionnaire for User Interface Satisfaction (QUIS; Chin, Diehl, & Norman, 1988). QUIS is a widely-used and well-researched usability questionnaire. It was adapted by adding extra questions about the sequencing of selecting, editing, and submitting patterns in SPEC. No questions were removed from the QUIS questionnaire. Participants were instructed to skip any questions they thought were not applicable to the interfaces they used. The questionnaire is presented in Appendix B. For the sake of clarity, the adapted QUIS is called the QUIS-a in this report. The interview was conducted after the participants finished the questionnaire. The interview followed a semi-structured format. During the interview, the test administrator solicited comments on these topics: the desired workflow for creating plans and specifications, whether the English representation of patterns were easy to understand, design ideas for organizing the pattern library, design ideas for integrating the SPEC and the TSD, design ideas for the Pattern Editor, other types of specifications/patterns that they think would be useful or that they would like to write. The complete list of interview questions is contained in Appendix C.

3.3 Outcomes

Of the six people that participated in the usability test, two participants completed the mission before 30 minutes elapsed. Two participants came close to completing the mission within 30 minutes. They had generated vehicle plans for the three vehicles and had created all the specifications but did not have vehicle plans that satisfied all specifications within the 30 minute timeframe. One participant had approximately half of the specifications written within the timeframe but the specifications were not satisfied. One participant did not use the model checker during the timeframe. This participant was trained on how to use the model checker just as others were and seemed to understand its purpose during training. During the mission task, he decided to spend time building

routes to accomplish the mission and did not create specifications in SPEC or use SPEC at all. His limited performance seemed to be due to an inability to work at a rapid pace in this task with this software, so perhaps he would have benefited from additional training. Nonetheless, this participant's data was included in the analysis because of his involvement with the model checker during training.

The QUIS-a data was analyzed using descriptive statistics and using the Kolmogrov-Smirnov non-parametric test. Each item of the questionnaire had a different textual anchor but no items were negatively coded so 10 was always a positive value and 1 was always negative value. Descriptive statistics showed that overall mean score was 6.53 (SD = 2.86). Average responses for each question with standard error bars are shown in Figure 7. Despite the small sample size, the Kolmogrov-Smirnov test found responses for 5 QUIS-a items were significant: 7, 8, 11, 24, 29. These items are the light colored bars shown in Figure 7. For item 7, characters on the computer screen were easy to read ($D(6) = 0.7$, $p < .01$). For item 8, highlighting on the screen simplified the task ($D(6) = 0.53$, $p < .05$). For item 11, the use of terms throughout the system was consistent ($D(6) = 0.53$, $p < .05$). For item 24, the system was reliable ($D(5) = 0.6$, $p < .05$; there were only 5 degrees of freedom because one participant did not respond to item 24). For item 29, the sequence of operations was clear ($D(6) = 0.53$ $p < .05$). In summary, all of the significant results were in the positive direction.

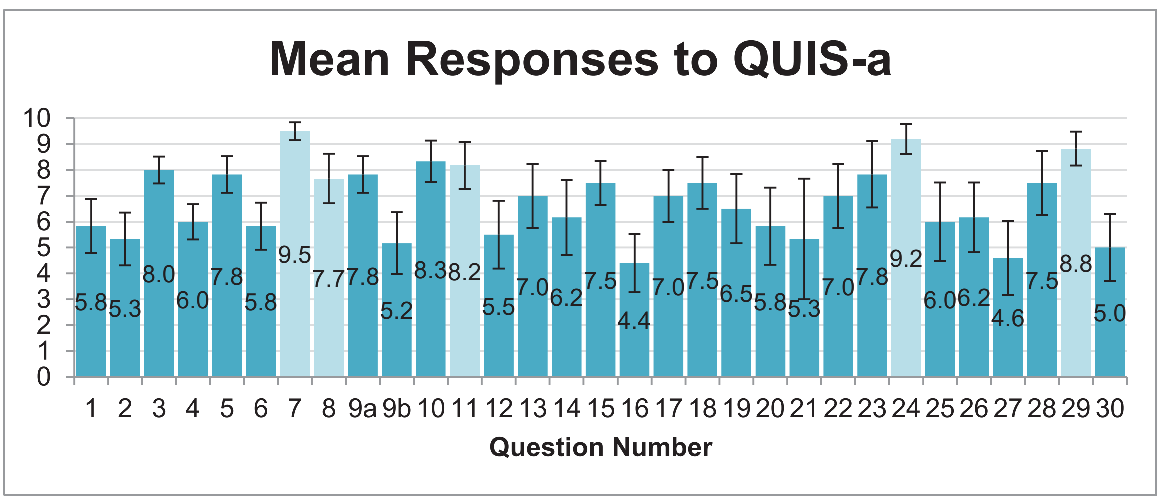


Figure 7. Average responses to QUIS-a questions with standard error bars. Each question had different anchors but a value of 10 was always positive and a value of 1 was always negative. Means are displayed numerically on each bar for convenience. Light-colored bars are significant based on Kolmogorov-Smirnov test (see text for explanation). Questions are presented in Appendix B

QUIS-a had space after each item for participants to supply comments about their response. A general theme was apparent for many participants and items. For item 11, use of terms throughout the system was inconsistent (1) or consistent (10), three participants commented that terms were confusing at times and terms used in SPEC did not match the terms they used to think about specifications. For item 17, learning to operate the system was difficult (1) or easy (10), and item 20, tasks can be performed in a straight-forward manner never (1) or always (10), participants provided additional comments about being confused by terminology and having difficulty understanding the patterns. Another topic of agreement in the free response comments was for items 9a and 9b; participants found the variable tree in the Edit Panel to be difficult to navigate. This topic was further explored in the semi-structured interviews, which are presented below. Most of the other comments related to VSCS and route planning on the TSD, which was not the primary focus of this study.

Participant's responses to interview questions were more varied than responses and comments on the QUIS-a. Interview responses are presented here in summary form for each interview prompt. Participants first were asked about their workflow strategy for completing the mission. In other words, did participants plan all the vehicle routes then build all the patterns, vice versa, or some interleaving of the two processes. Most participants (4 of 6) interleaved the two processes. One participant planned all the routes first and one participant built all the patterns first. Follow-up questions on workflow strategy focused on whether the designs of VSCS and SPEC suited their desired strategy. Participants were divided between yes (3 of 6), no (2 of 6), and somewhat (1 of 6). These responses were spread out across the different workflow strategies and did not appear to be related to strategy used. Participants mentioned that the interface is well-suited to interleaving, which was the most common strategy used. One participant felt the design forced them into interleaving when he would have preferred to build all the patterns before beginning route planning. Participants were asked if it was clear how to fix issues that were flagged by the model checker. In general, participants felt that the model checker flag next to the specification was not clear by itself and more feedback about the failure or even a suggested solution to the failure would improve the interface.

Participants mentioned that much of the ambiguity in addressing violated specifications was troubleshooting the source of the problem: whether there was a mistake creating the specification in SPEC (wrong pattern selected, wrong variables, etc.) or whether the UAV plans didn't satisfy a set of correctly expressed specifications.

Participants were asked if they found the patterns in the pattern library easy to understand. Responses varied from participant to participant. Three participants said that some patterns were easy to understand and other patterns were not easy to understand, but the examples each participant used did not agree. Two participants said that the patterns were easy to understand. The one pattern that more than two participants agreed was confusingly worded was the bounded response pattern, which contained the phrase "after at most" *x* units of time. Participants were asked if they had any wording suggestions for the patterns in the pattern library. Most participants did not have any suggestions and acknowledged the difficulty of finding a wording that was easy to understand and also unambiguous. Two participants suggested adding a graphical representation of each pattern, if possible, to provide another explanation of the relationship the pattern represents.

Participants were asked to provide design ideas, if they had them, for different components of SPEC. First, participants were asked about the pattern library's organization. No consensus arose from interview responses. Two participants generally liked the organization. One participant suggested having a keyword search but another participant mentioned that searching would not be effective. Interestingly, one participant did suggest typing the pattern and having it translated to temporal logic, which was a previously explored approach in this project (e.g., Rothwell et al., 2013). Next, participants were asked about design ideas for integrating SPEC and the control station's map, i.e. the TSD. Participants agreed that integration between SPEC and the TSD would be beneficial but were not sure how to accomplish it. One participant suggested a timeline display to go underneath the TSD. One participant suggested being able to create minimum duration patterns in SPEC from the loiter exit criteria in the vehicle planning pop-up window (i.e., "quick edit" window) on the TSD. Another participant wanted to see the violations flagged by the model checker represented on the map as more informative feedback.

Last, participants were asked about design ideas for the Edit Panel. Participants agreed that the variable tree is hard to navigate and too large when all the levels are expanded. They requested that the variable tree be improved, but differed on how to improve it. One participant suggested closing all the lower tree levels. One participant suggested having a "collapse all" button. Another participant closed all the lower levels manually each time. Three participants wanted alternative methods to enter variable information, either typing or from drop down (i.e., "combo") boxes. Three participants mentioned having the selected variable advance to the next variable automatically or by pressing the tab key on the keyboard, in addition to mouse click selection.

Participants were asked what tasks/missions would be helped by the SPEC interface and model checking capability. A number of participants suggested SPEC would be a benefit to missions that are well defined and not dynamically changing, as well as missions that were not time critical because of the time it takes to enter in specifications. The mission types recommended were: UAV missions, ground mission surveillance, surveillance in general, base defense, strike/attack/combat, aerial refueling, communication relay, and moving target indication.

Participants were asked if there were any other types of patterns that would be useful. Also, this question was restated to ask if participants had any additional goals, objectives, or constraints to convey to the automation that they were not able to convey using these patterns. Two participants asked for combinations of specifications. Three participants wanted to be able to use multiple UAVs in one pattern (e.g., Bat-21, Bat-22, and Bat-23 avoid ROZs or all UAVs avoid ROZs).

3.4 Conclusion

A usability study was conducted on an initial SPEC display. Participant feedback was generally positive, showing the promise of these kinds of technologies. The usability study also generated many ideas for further development and improvement of the SPEC tool, however, only a few topics approached or achieved a consensus: participants expressed a desire for more feedback from the model checker, participants found the pattern wording difficult to understand at times, and participants found the Edit Panel's variable tree hard to navigate. Below, we report additional research that was conducted to address these issues.

Feedback could be increased through providing an explanation of where and how the vehicle plans violated the specifications. Participants desired more feedback from the model checker so they could determine more easily if the specification violation was due to a problem in their UAV plans or a problem in the specifications they created. This troubleshooting process for verifying UAV mission plans largely parallels the analysis of model checking results in the original domain of model checking, i.e. hardware and software verification. Consider the UAV plans to be our instance of a system *model* and the specifications to be our instances of system *properties*. Baier and Katoen (2008) discuss how "whenever a property is falsified, the negative result may have different causes" (p. 13). There may be a *modeling error* (the model is not a valid representation of the system), a *design error* (the system has a flaw which has been discovered), or a *property error* (the specification checked does not reflect the informal requirement). Participants of their own accord realized the possibilities for design errors (i.e., problems with their plans) and property errors (i.e., problems with their specifications). Determining the cause of a negative result is not always simple, and better feedback would most likely enhance the distinctions between design errors and property errors. We note that this type of "error explanation" in model checking is an open area of research and that, as discussed by Groce, Chaki, Kroening, and Strichman (2005), human understanding of errors is a challenging problem that is unlikely to be resolved through mathematical proofs from formal methods approaches. We also note that the usability study did not come across any participants who encountered or suspected modeling errors, though these errors might be a real possibility for a fielded system and the task of investigating and identifying these errors using SPEC should be considered in future work.

Pattern understanding could be improved through two means: improving pattern wording or adding an additional representation of the temporal relationship that supplements the pattern wording, such as a graphical representation or auxiliary textual description. Improvements to pattern understanding should reduce the number of initially incorrect specifications (i.e., property errors) and facilitate differentiating between design errors and property errors when they occur. The variable tree in the Edit Panel could be

improved by altering the hierarchical expand and collapse behavior, providing alternate input methods to the tree, and automatic advancement once a variable has been selected.

4.0 INTERFACE DESIGN CHANGES

The usability test identified three areas of improvement in the SPEC design: understanding of the patterns, the interaction with the Edit Panel, and a desire for the model checker to provide explanations of errors or automatically repair erroneous plans. For a second version of SPEC, we addressed two of these areas: understanding of the patterns and the interaction with the Edit Panel. Due to technical complexity and project constraints, we were not able to address model checker feedback for the second version of SPEC. SPEC's changes are detailed below.

Pattern Understanding

To increase understanding of patterns, we did two things. First, we changed the wording of many of the patterns in an effort to make them more readable while trying to avoid ambiguity in the temporal relationships they express. Second, we developed a supplemental description of the pattern but with everyday circumstances, to try to further convey the logical relationship in a natural intuitive example (Table 1).

Table 1. Examples of the Descriptions, 'everyday' examples created to increase pattern understanding

Pattern	Description
`Before the first occurrence of {Y}, {VAR} happens at least once`	Before Bob leaves the grocery store, Bob checks his grocery list one or more times.
`Across all time, {VAR1} must happen before {VAR2} can happen.`	It's always the case that you must start a car before you can drive it anywhere.
`After the first occurrence of {X}, whenever {VAR} happens it holds for at least {C}.`	After Jill moved to the beach, whenever her parents visit her they stay for at least 2 weeks.

Edit Panel Interaction

To improve navigation of the Edit Panel, the majority of the interaction changes focused on the tree viewer's behavior. Many complaints focused on the length of the tree list due to the many variables in the NuSMV model and on the expand/collapse behavior of the tree viewer. We did not want to remove variables from the model and so kept the tree length the same in terms of number of items, but tried to reduce its visual length by creating new collapsing behaviors for tree levels. We added a "collapse all" button to provide a quick method to clean up the variable tree. In addition, we modified how actions at a higher (i.e., parent) level of the tree affect the lower (i.e., child) levels of the tree. A conventional tree viewer requires each level of the tree to be manually collapsed and expanded. This means that when a parent level is collapsed or expanded it has no effect on its children levels. The tree viewer behavior was modified so that any collapse action would also collapse all children of that level. This meant that next time the parent level was opened, the children would always be in the collapsed state. We did not modify the expand behavior of the tree. In other words, when a parent level was expanded there was no change in the state of its children levels (they would remain collapsed from the previous inheritance of the collapse action).

The Edit Panel was also criticized for the heavy mouse usage in selecting which variable to assign (at the top of the panel) and which variable value the user wants to use (in the variable tree). To address these issues, we had the selected variable (i.e., the variable that will be assigned when something is selected in the tree) automatically advance to the next variable in the pattern after a selection was made. Previous design of SPEC required users to manually return to the list of variables within the pattern and change which variable was selected via mouse click. This one change eliminated a lot of mouse travel back and forth between the variable list and the variable tree. In addition, for the selection of time in the tree we added a field of hours, minutes and seconds (hh:mm:ss) that can be modified by the keyboard up and down arrow keys.

One change was made to the Edit Panel that was not motivated by the usability study but rather by a design goal of furthering the integration of the temporal patterns and the spatial representation on the map of the control station. We added buttons to the

variable tree that allow the user to select objects (e.g., Points, Areas, Restricted Operating Zones) from the TSD on the other screen.

Model Checker Feedback

Improving the feedback from the model checker was not accomplished for the second iteration of the design. Two different ways to increase model checker feedback were entertained but not implemented due to the level of effort required. First, for certain types of specification patterns, it is possible to indicate where on the map the specification is violated. Second, it is possible to present a timeline display that shows the state changes of the variables contained in the failing specification. Some other approaches to error explanation in the context of software debugging are discussed by Groce, Chaki, Kroening, and Strichman (2005). Though these feedback concepts were not implemented at the time, they are candidate topics for future research and development.

5.0 EXPERIMENT

After refining the user interface in response to the feedback obtained in the usability study, we conducted an experiment to test if verification tools can improve the accuracy of mission plans. The primary experimental manipulation was the presence or absence of SPEC during a mission planning vignette. The experiment was designed to measure the impacts of SPEC on mission planning accuracy, mission planning time, and mental workload. We hypothesized that using SPEC would increase mission planning accuracy but would also increase mission planning time, as it would likely take longer to do mission planning and checking than just mission planning alone. We hypothesized that using SPEC might also increase mental workload, since it requires users to write specifications and possibly to revise mission plans after receiving feedback from SPEC.

5.1 Method

Participants

Twelve individuals (four female) volunteered to participate in this experiment. These individuals were drawn from the population of scientists and engineers at Air Force Research Laboratory’s Human Performance Wing. None of these participants had prior experience with SPEC. All participants had normal or corrected-to-normal vision

and reported normal color vision. Some participants had previous experience with prototype UAV ground control stations including VSCS. All participants provided informed consent in accordance with the Institutional Review Board of the Air Force Research Laboratory's Human Performance Wing.

Materials

The experiment was conducted in a quiet office setting using a desktop computer with 2 monitors 24" in size at a resolution of 1920 x 1200 (Dell 2408WFPb; Round Rock, TX). VSCS was presented across the 2 monitors with the TSD on the left monitor (Figure 2). When SPEC was present, it was presented on the right monitor. When SPEC was absent, the right monitor was blank. Two mission planning scenarios were developed based on the scenario used in the usability study (shown in Appendices D and E). They each had eight mission objectives that were similar in content and approximately equal in difficulty. The two scenarios were counterbalanced with display conditions during the experiment to control for any unintended differences between the scenarios. Workload was measured using the NASA Task Load Index (NASA-TLX; Hart & Staveland, 1988), which was completed using a computerized version of the questionnaire. The pairwise comparison aspect of the test was completed once, at the end of the experiment, and the weights applied to all of that participant's responses. The QUIS-a was administered on paper, which was unchanged from the usability study described above. In addition, we performed a semi-structured interview slightly modified from what was used in the usability study. It was shortened for the sake of time and one question was added to solicit feedback on the perceived benefit and value of SPEC.

Procedure

The experimental design was a 2 x 2 design with the within-subject factors display condition (SPEC absent or present) and scenario (A or B). Each participant performed two trials, one with SPEC and one without SPEC. The combination of display condition and scenario was counterbalanced across participants, making the interaction between-subjects. (The interaction was not a focus of this study and attempts were taken to equate the two scenarios.) Also, the order of conditions was counterbalanced across participants to control for any order effects. Upon arriving to the lab, participants read informed consent documents and asked any questions. After providing their consent, they

were assigned to one of four sequences depending on the counterbalanced combinations of display conditions and scenarios. The sequence of training differed depending on whether SPEC was present or absent in the first trial.

If SPEC was absent in the first trial, participants were trained on how to create mission plans using VSCS. Training consisted of: an explanation of the features and various methods of mission planning in VSCS, a demonstration of mission planning, and hands-on practice mission planning. Training was completed when the participants demonstrated they could create a number of specific vehicle plans that exercised many different aspects of mission planning features in VSCS. After completion of training, participants completed the first trial. All trials were self-paced and participants were instructed to work on the mission plan until they were satisfied they had met all the objectives. Accuracy was emphasized over speed. The completion time was recorded by the experimenter using a stop watch. After trial completion, participants filled out the NASA-TLX. Then prior to the second trial, participants were trained on how to use SPEC. SPEC training consisted of: an explanation of the features of the SPEC tool, a discussion of the types of patterns available, a demonstration of writing specifications with SPEC, hands-on practice using SPEC to create specifications that exercise many different aspects of the tool, and practice using it in a scenario—checking those specifications against mission plans, receiving feedback from SPEC, and addressing problems both in the mission plan and the specification. After training, participants completed the second trial followed by the NASA-TLX questionnaire and pairwise comparison selections. If SPEC was present in the first trial, participants were trained on creating mission plans in VSCS followed by training on how to use SPEC. Then, they completed the first trial and NASA-TLX questionnaire followed by the second trial and NASA-TLX questionnaire and the pairwise comparison.

After the completion of both trials (irrespective of order), participants completed the QUIS-a in regards to the SPEC tool. They were asked specifically to not address VSCS in their responses, as that was not the focus of this research. Once QUIS-a was completed, the semi-structured interview was conducted. The whole session lasted 3 hours and 45 minutes on average. At every transition between segments of the experiment, participants were prompted to take a break, if they desired.

5.2 Results

The following results showed that the presence of SPEC led to increased mission planning accuracy as well as increased mission planning time, but there was not a significant increase in subjective workload. The scenarios were not significantly different in planning accuracy, however Scenario A took longer than Scenario B, and Scenario A was rated higher in overall workload and temporal demand than Scenario B. Additional analyses of participants' interactions with SPEC using quantitative, questionnaire and interview methods showed that generally participants liked SPEC and found it to be useful.

Experimental Results

We calculated mission planning performance scores for each block as the proportion of mission objectives that were met by the vehicle plans participants created. A preliminary review of descriptive statistics (mean, SD) suggested that outliers may be present in the data. We used a quartile-based outlier identification technique, where a threshold for outliers was set based on the 1st and 3rd Quartile ±1.5 times the inter-quartile range (IQR). We performed this calculation for each treatment cell individually. That is, we did this for the SPEC absent data and again for the SPEC present data. The lower threshold value for the SPEC absent data was 0.5625 and Participant #5's score of 0.25 was below that value. The lower threshold value for the SPEC present data was 0.75 and Participant #8's score of 0.625 was below that value. Both of these outlier values were obtained on the first trial suggesting perhaps these subjects needed additional training or had difficulty comprehending the instructions, despite achievement-based training criteria and ample opportunities for questions (though this need for more training was not apparent in the total data, as indicated by the insignificant test for order effects below). To preserve the balanced nature of our design, we then sought to impute data values for these two outliers. Some researchers advocate imputing the outlier values with the respective treatment means after removal of the outliers, but a more conservative practice (that can actually increase Type-I error) is to use the overall mean after removal of the outliers (Tabachnick, Fidell & Osterlind, 2001). We used the more conservative practice in this instance, and the overall mean value used for imputation was 0.8579.

A repeated-measures analysis of variance (ANOVA) of performance scores was done to test for order effects, including display condition and scenario as factors. No order effects were found ($F(1, 9)=0.35$, $p = .57$). Scenario, even though it was crossed with display conditions, was analyzed in order to check that we met our intention of having scenarios that were similar in difficulty. The mean mission planning performance for Scenario A was 0.78 and for Scenario B was 0.86. This difference was not significant, $F(1, 9) = 1.17$, $p =.31$. The main effect of display condition was significant, $F(1, 9)=5.68$, $p < .05$, $\eta_p^2 = .39$. SPEC present trials had higher performance compared to SPEC absent trials (0.915 vs 0.801, respectively).

Mission planning time was analyzed after log transform of the completion times, to reduce the positive skew in the distribution of values. We conducted a repeated-measures ANOVA with display condition, scenario, and order as factors. There was no order effect, $F(1, 9)=0.01$, $p < .98$. There was a main effect of display condition ($F(1, 9) = 42.58$, $p < .001$, $\eta_p^2 = .83$). When SPEC was present, participants took longer to plan (M=3.36 log seconds, SD = .17) than when SPEC was absent (M = 2.99 log seconds, SD = .21). The units of log seconds can be difficult to interpret, so we have converted the means into seconds for comparison; planning with SPEC present was 2290.87 seconds (38 minutes, 11 seconds) whereas planning without SPEC was 997.24 seconds (16 minutes, 37 seconds). There was a main effect of scenario as well ($F(1, 9) = 10.01$, $p < .05$, $\eta_p^2 = .53$). The mean (SD) planning time for Scenario A was 3.26 (.23) log seconds and for Scenario B was 3.08 (.27) log seconds. Expressed in seconds, Scenario A times averaged 1819.7 seconds (30 minutes, 20 seconds) and Scenario B times averaged 1202.3 seconds (20 minutes, 2 seconds).

NASA-TLX scores of overall workload and each subscale were analyzed using repeated-measures ANOVA with factors of display condition, scenario, and order (Figure 8). No order effects were present in the data (all $p > .13$), so those results will be omitted for brevity. For overall workload, there was no difference between display conditions (41.81 for both SPEC absent and present), $F(1, 9)= 0.00$, $p = 1.00$. There was a main effect of scenario ($F(1, 9) = 5.35$, $p < .05$, $\eta_p^2 = .37$), with Scenario A being perceived as more workload than Scenario B (47.31 and 36.31, respectively). For the mental demand subscale, there was no main effect of display condition ($F(1, 9) = 0.27$, $p = .62$). SPEC

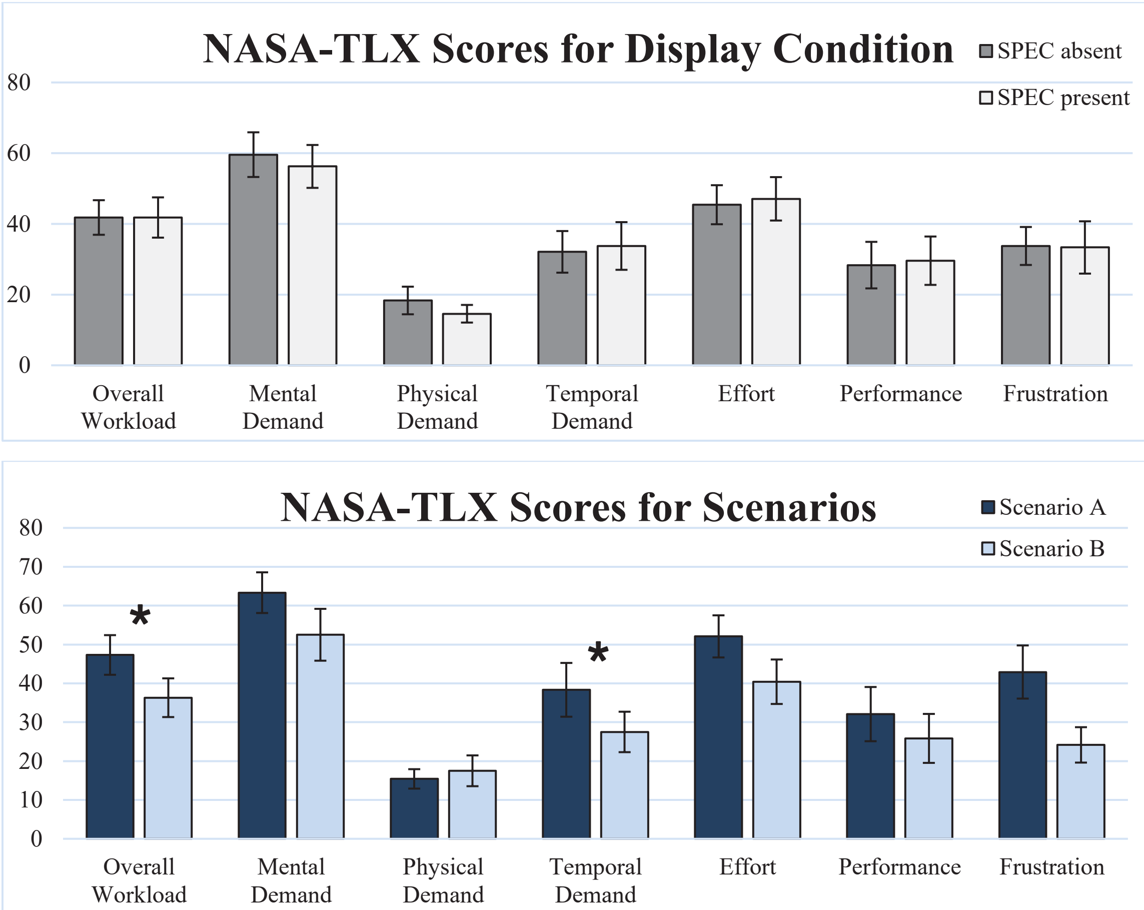


Figure 8. Summary of results for NASA-TLX overall workload and individual subscales. The top panel shows the tests of display condition main effect. The bottom panel shows the test of the Scenario main effect. Significant differences at $p < .05$ level are marked with *. See text for detailed explanation.

absent was slightly higher than SPEC present but this was not significantly different (59.6 and 56.25, respectively). There was no main effect of scenario either, ($F(1, 9) = 2.85$, $p = .13$). Similarly, for the physical demand subscale there was no main effect of display condition ($F(1, 9) = 2.33$, $p = .16$). SPEC absent was slightly higher than SPEC present but this was not significantly different (18.33 and 14.58, respectively). There was no main effect of scenario, ($F(1, 9) = 0.72$, $p = .42$). For the temporal demand subscale, there was no main effect of display condition ($F(1, 9) = 0.14$, $p = .72$). There was a significant main effect of scenario ($F(1, 9) = 5.94$, $p < .05$, $\eta_p^2 = .40$), Scenario A was higher than Scenario B (38.33 and 27.50, respectively). For the effort subscale, there was no main effect of display condition ($F(1, 9) = 0.06$, $p = .82$) or for scenario ($F(1, 9) = 2.82$, $p =$

.13). For the performance subscale, there was no main effect of display condition ($F(1, 9) = 0.11$, p = .75) or for scenario ($F(1, 9) = 2.78$, $p = .13$). For the frustration subscale, there was no main effect of display condition ($F(1, 9) = 0.028$, $p = .96$). There was no a main effect of scenario ($F(1, 9) = 4.20$, $p = .07$), though Scenario A was rated more frustrating than Scenario B (42.92 and 24.17, respectively). In summary, there were no main effects of display condition and Scenario A was significantly higher workload than Scenario B for overall workload and temporal demand (and perhaps a trend for frustration).

Results on Using SPEC

To further investigate participants' performance with SPEC, we analyzed the accuracy of writing specifications in SPEC for each scenario mission objective. The overall accuracy for specification writing was 81.25% (SD = 21.8). The data indicated that one type of objective was problematic. The mission objective with the lowest accuracy for Scenario A was objective 7 (33%) and for Scenario B was objective 6 (50%), which were both objectives requiring overwatch of the Convoy while it was in a particular location. This suggests that the pattern that expressed this objective was probably difficult to understand and/or difficult to recognize that it was the appropriate pattern.

One requirement for the utility of the model checker is correct specifications. Without participants writing correct specifications, the model checker feedback may cause confusion because the feedback is not entirely due to the plan (recall the earlier distinction between a *design error* and a *property error* on p. 19). In other words, the model checker could return a negative result when an incorrect specification was submitted on a correct plan or it could return a positive result when an incorrect specification was submitted on an incorrect plan. Therefore, we suspected that participants' performance in writing specifications may be correlated with their performance in mission planning. To test this, we looked within blocks when SPEC was present and performed a correlation between mission planning accuracy and specification writing accuracy. This was significant $r = .71$, $p < .001$ (95% CI = .23 - .91). This should be interpreted with some caution, however, as this is not a strong test for causality. An alternative explanation is that participants who perform well in mission planning also

performed well in writing specifications, so the participant might have been the cause of this relationship rather than having a correct specification to check against.

Additional analyses focused on the subjective data collected in the QUIS-a questionnaire and interview procedures. The QUIS-a responses were analyzed in two ways: to look within the ratings of experiment participants and to compare between participants in the Usability study and the experiment. The questionnaire items are on a 10-pt Likert scale from 1 to 10. Looking within the experiment, responses were generally positive (M = 8.01, SD = 1.58) and above the mid-point of 5.5. The lowest rated item was question 4, which had a mean of 6.58. Question 4 asked about overall reactions to the software with the anchors: difficult (1) – easy (10). The second lowest item was question 2, which had a mean of 7.08. Question 2 asked about overall reactions to the software with the anchors: frustrating (1) – satisfying (10). The third lowest item was question 31, which had a mean of 7.17. Question 31 asked "while performing the sequence of pattern building I made: many mistakes (1) – no mistakes (10)." A few items were highly rated with means above 9: question 23 (system speed), question 24 (system reliability), and question 35 (system should be used infrequently (1) – frequently (10)).

To tentatively assess the changes made to SPEC after the usability study, we compared the QUIS-a scores from the experiment to those from the usability study, which used different versions of the SPEC display. This tentative examination used descriptive statistics and did not perform any statistical tests. Each questionnaire item was compared by taking the difference between the experiment rating and the usability rating for each item, such that positive values meant the experiment (and refined version of SPEC) was rated higher than the usability study (initial version of SPEC) (shown in Figure 9 below). The mean difference was 1.75 (SD = 1.43), suggesting the participants' ratings of the SPEC tool were generally higher in the experiment than in the usability study.

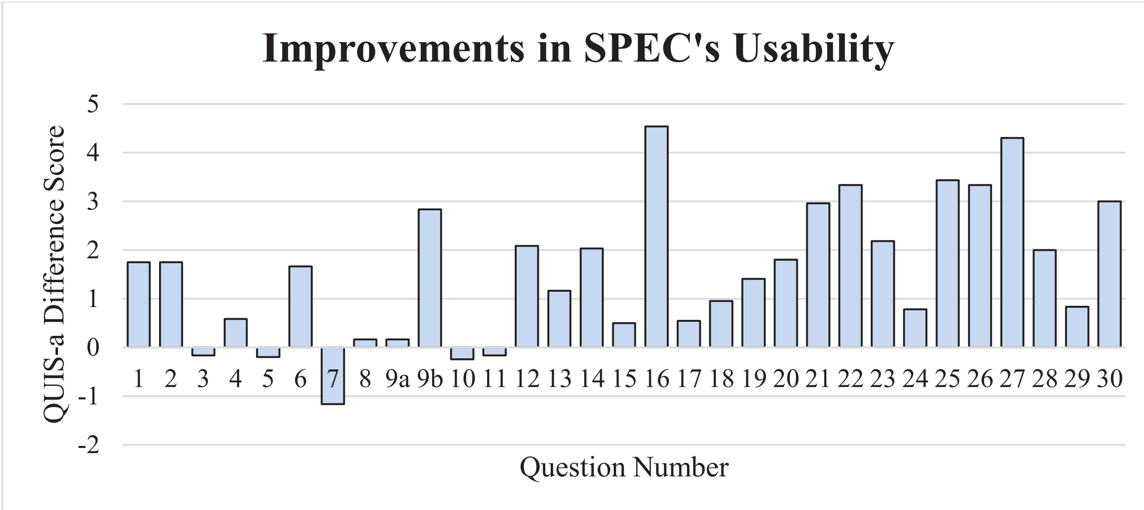


Figure 9. A comparison between QUIS-a scores for the SPEC version used in the experiment and the SPEC version used in the usability study. Positive values occurred when the experiment ratings were higher than the usability study ratings

Participant's responses to interview questions were compiled and reviewed for agreement amongst participants. Interview responses are presented here in summary form for each interview prompt. Participants first were asked about their workflow strategy for completing the mission (i.e., if they planned first then wrote patterns, vice versa, or interleaved the processes). Half of participants (6 of 12) interleaved the two processes and the remaining used a more serial strategy. Follow-up questions on workflow strategy focused on if the designs of VSCS and SPEC suited their desired strategy. Half of participants reported (6 of 12) that their workflow was impacted by the design that a specific UAV had to be selected. Participants were unable to use a generic UAV such as "any UAV" and the Boolean logic of "and" and "or" was not available. This influenced all six of them to decide which vehicle to use, make the route, and then write the specification. (It was possible for participants to have written the specifications after their decisions but prior to the route planning, and this approach was adopted by three other participants who created all the specifications first).

Participants were asked if it was clear how to fix issues that were flagged by the model checker. As was found in the usability study, participants felt that the pass/fail feedback from the model checker was not clear by itself and more feedback about the failure or a suggested solution to the failure would improve the interface. Three participants had failures related to timing and or altitude and particularly desired the

model checker to provide some information about the direction of the problem (too early/late, too high/low).

Participants were asked if they found the patterns in the pattern library easy to understand. Most participants (8 of 12) responded that the patterns were easy to understand or would be easy with more familiarity/practice. Regarding the everyday examples that were added to the patterns after the Usability study, 4 participants used them and found them helpful. The other nine participants did not use the examples and a suggestion was made that examples would be more helpful if they were in the UAV domain. Participants were asked if they had any wording suggestions for the patterns in the pattern library. Most participants did not have any suggestions, though a few phrases were viewed negatively by participants, e.g., “first occurrence” and “across all time”.

Participants were asked to provide design ideas, if they had them, for different components of SPEC. First, participants were asked about the pattern library’s organization. Participants disagreed on the categorization of the patterns. Some participants found the categories and subcategories helpful and informative, while others did not and tended to be frustrated by manually looking. Three participants requested a search capability so that they could search to narrow down alternatives (adding a search capability was also mentioned in the Usability study). Two participants suggested changing how the variables in the pattern are colored so that each variable is always the same color. Two participants reported that the highlighting on the pattern library panel was confusing. They mistakenly thought it was highlighting under the selected row rather than highlighting on the selected row.

Next, participants were asked about design ideas for integrating the SPEC and the control station’s map, the TSD. Two participants requested that specifications that failed the model checker should be displayed on the map with some additional information about why they failed. One participant expressed concern that additional feedback on the map might increase the clutter in an undesirable manner. Another participant suggested that the independence of the two displays contributed to the idea of validation, and made an analogy to two-man programming.

Last, participants were asked about design ideas for the pattern Edit Panel. Some participants (4 of 12) appreciated the collapse all button, yet others (4 of 12) still had

difficulties with the length of the variable tree. Two participants noted that when the tree was expanded, the vertical scroll window would jump position and often the newly expanded content was off the bottom of the screen. This was frustrating as they always had to manually scroll down to view the content that they were clearly attempting to access.

Participants were asked if there were any other types of patterns that would be useful. Responses were quite varied but there were two sources of agreement. Participants reiterated the desirability of being able to specify “any UAV” or “all UAVs.” The other point of agreement in these responses was to allow clock time in specifications, for example clock time could appear in the specification initial clause (i.e., “before Y time”, “after X time”, or “between X and Y time”). In addition, participants had many other suggestions however there was little overlap between them. One participant wanted to specify absolute altitude in addition to altitude bands. Another participant was considering mission checks during the execution of the mission and wanted to specify and check for how long / far the UAVs could go without breaking the plan. Another participant suggested adding a way to chain or nest specifications in order to handle more complex missions. Continuing the complexity discussion, this participant also suggested adding variables about vehicle payloads, such as different sensor types, so that specifications could be made to require a specific payload type rather than a specific UAV.

The final question solicited participants’ perceptions about the value of the SPEC tool. All participants liked the tool, thought it added value and provided additional validation of their plans. Furthermore, participants who used SPEC first felt that they were not as confident in their plans on their second trial, when they did not have SPEC.

5.3 Discussion

We found that there is a trade-off between the speed and accuracy of planning associated with a model checking tool such as SPEC. When participants used SPEC, they were more accurate in creating plans that accomplished the mission (91.5% with SPEC vs 80.1% without SPEC) but they also took substantially more time to complete their planning to use SPEC and troubleshoot their plans (an additional 20 minutes). In a typical speed-accuracy trade-off situation, additional time spent will tend to result in increased

accuracy. Here, we do not know what performance would have been like if participants had spent 20 extra minutes planning when they did not have SPEC. The extra time could have also improved performance, however it may not have because participants already were instructed to emphasize accuracy over speed. When considering the extra time spent using SPEC, it is important to recall that using SPEC did not contribute to increased perceptions of workload. Nonetheless, both effective and efficient mission planning capabilities are being sought, thus additional research is warranted to investigate how to reduce the time needed to use SPEC. Some possible approaches to reduce the time required are: provide additional training and familiarization with the tool, modify the interface to improve interactions, or some combination of the two.

With regard to the scenarios, the results indicated that our attempts to balance the difficulty of the scenarios were not entirely successful. Scenario A appeared to be more challenging than Scenario B. Participants took longer with Scenario A than Scenario B and rated Scenario A as more demanding than Scenario B. The Scenario did not significantly affect planning performance, and the scenario was crossed with the availability of the SPEC tool so the interpretation of the SPEC tool results was not affected.

6.0 CONCLUSION

This report described the development and testing of SPEC, a model checking tool for verifying UAV mission plans. This tool was developed in anticipation of the increases in mission complexity and automation that will be the result of the Department of Defense's vision to increase the use and capability of unmanned systems. A key aspect of SPEC is providing the user a method to communicate their mission objectives, tasks and constraints in man-readable form to the model checking software so the model checker can perform verification of the mission. SPEC was developed so that UAV operators could create mission specifications and use model checking software without having to learn temporal logics, instead accomplishing this through temporal patterns. The first iteration of SPEC was subjected to a usability evaluation which led to identification of a number of areas for improvement. We identified three problem areas to focus on: participants desired more feedback from the model checker, participants found some of the patterns difficult to understand, and participants had trouble navigating in the

Edit Panel's variable tree. We addressed the latter of these two areas by developing a second, more refined version of the SPEC tool. This second version of SPEC was experimentally tested to investigate the effects of verification tools like model checkers on pre-mission route planning. The experiment found that model checking tools can increase the accuracy of UAV mission planning without adding additional cognitive workload, but these tools do require additional time to use. Participants reinforced these quantitative findings through reporting that they found the model checking tool to be a very valuable. Further research could focus on reducing the time required through additional training and/or improvements to SPEC, both with respect to writing specifications and presenting model checker feedback. Further research could also focus on error explanation for model checking to help improve feedback.

7.0 REFERENCES

Baier, C. & Katoen, J. (2008). *Principles of model checking*. Cambridge, MA: MIT Press.

Cavada, R., Cimatti, A., Jochim, C., Keighren, G., Olivetti, E., Pistore, M., Roveri, M., & Tchaltsev, A. (2010). NuSMV 2.5 User Manual, FBK-IRST: Fondazione Bruno Kessler / Istituto Ricerca Scientifica e Tecnologica, Trento, Italy, Retrieved from: http://nusmv.fbk.eu/NuSMV/userman/v25/nusmv.pdf, 2010.

Cimatti, A., Clarke, E., Giunchiglia, E., Giunchiglia, F., Pistore, M., Roveri, M., Sebastiani, R., & Tacchella, A. (2002). "NuSMV 2: An opensource tool for symbolic model checking." *Computer aided verification* (pp. 359-364). Springer Berlin Heidelberg.

Chin, J. P., Diehl, V. A., & Norman, K. L. (1988). Development of an instrument measuring user satisfaction of the human-computer interface. In *Proceedings of the SIGCHI conference on Human Factors in Computing Systems* (pp. 213-218). New York: ACM.

Doherty, P., Kvarnstrom, J., & Heintz, F. (2009). A temporal logic-based planning and execution monitoring framework for unmanned aircraft systems. *Autonomous Agent Multi-Agent Systems, 19*, 332–377.

Dwyer, M., Avrunin, G., & Corbett, J. (1999). Patterns in property specifications for finite-state verification. *Proceedings of the 21st International Conference on Software Engineering,* (pp. 411–420). ACM.

Feitshans, G., Rowe, A., Davis, J., Holland, M., & Berger, L., (2008). Vigilant Spirit Control Station (VSCS): The face of COUNTER. *Proceedings of the AIAA Guidance, Navigation, and Control (GNC) Conference*. doi:10.2514/6.2008-6309

Hart, S.G. & Staveland, L.E. (1988). Development of NASA TLX (Task Load Index): Result of empirical and theoretical research. In P. A. Hancock & N. Meshkati (Eds.), *Human Mental Workload* (pp. 239-250). Amsterdam: North Holland Press.

Groce, A., Chaki, S., Kroening, D. & Strichman, O. (2005). Error explanation with distance metrics. *International Journal on Software Tools for Technology*, *8*(*3*), 229-247.

Humphrey, L. (2012). Model checking UAV mission plans. *Proceedings of the AIAA Modeling and Simulation Technologies (MST) Conference*. doi: 10.2514/6.2012-4723

Humphrey, L., & Patzek, M. (2013). Model checking human-automation UAV mission plans. *Proceedings of the AIAA Guidance, Navigation, and Control (GNC) Conference*. doi: 10.2514/6.2013-5183

Humphrey, L., Wolff, E., & Topcu, U. (2014). Formal specification and synthesis of mission plans for unmanned aerial vehicles. *Proceedings of the AAAI Spring Symposium*. ~~doi:~~

Kim, B., & Humphrey, L. (2014). Satisfiability checking of LTL specifications for verifiable UAV mission planning. *52nd Aerospace Sciences Meeting, AIAA SciTech Forum*, (AIAA 2014-0793) http://dx.doi.org/10.2514/6.2014-0793.

Klein, G., Woods, D. D., Bradshaw, J. M., Hoffman, R. R., & Feltovich, P. J. (2004). Ten challenges for making automation a 'team player' in joint human-agent activity. *IEEE Intelligent Systems, 19*(6), 91–95.

Konrad, S. & Cheng, B. (2005a). Facilitating the construction of specification pattern-based properties. *13th IEEE Int'l Requirements Engineering Conference* (RE'05).

Konrad, S. & Cheng, B. (2005b). Real-time specification patterns. *Proceedings of the 27th Conference on Software Engineering* (ICSE'05).

Kress-Gazit, H., Fainekos, G., & Pappas, G. (2008). Translating structured English to robot controllers. *Advanced Robotics, 22*, 1343–1359.

Rothwell, C., Eggert, A., Patzek, M., Bearden, G., Calhoun, G., & Humphrey, L. (2013). Human-computer interface concepts for verifiable mission specification, planning, and management. *AIAA Infotech@Aerospace*, AIAA 2013-4804.

Rowe, A., Liggett, K., & Davis, J. (2009). Vigilant Spirit Control Station: A research testbed for multi-UAS supervisory control interface technology. *Proceedings of the 15th International Symposium of Aviation Psychology*.

Stanard, T., Bearden, G., & Rothwell, C. (2013). A cognitive task analysis to elicit preliminary requirements for an automated UAV verification & planning system. In *Proceedings of the Human Factors and Ergonomics Society Annual Meeting* (Vol. 57, No. 1, pp. 210-214). SAGE Publications.

Tabachnick, B. G., Fidell, L. S., & Osterlind, S. J. (2001). *Using multivariate statistics* (4th Ed.). Allyn and Bacon: Boston, MA.

8.0 Appendix A
Patterns in SPEC's Pattern Library

Occurrence

<u>Absence</u>

Across all time, {VAR} never happens

Before the first occurrence of {Y}, {VAR} never happens

After the first occurrence of {X}, {VAR} never happens

Between the occurrence of {X} and {Y}, {VAR} never happens

<u>Universality</u>

Across all time, {VAR} is always happening

Either {VAR1} or {VAR2} or both are happening at all times

{VAR1} is equal to {VAR2} at all times

{VAR1} always does not equal {VAR2} at all times

{VAR1} is always less than {VAR2} at all times

{VAR1} is always greater than {VAR2} at all times

Before the first occurrence of {Y}, {VAR} is always happening

After the first occurrence of {X}, {VAR} is always happening

<u>Existence</u>

Across all time, {VAR} happens at least once

Eventually, {VAR1} and {VAR2} both happen at the same time

Before the first occurrence of {Y}, {VAR} happens at least once

After the first occurrence of {X}, {VAR} happens at least once

Between the occurrence of {X} and {Y}, {VAR} happens at least once

Order

<u>Precedence</u>

Across all time, {VAR1} must happen before {VAR2} can happen

Before the first occurrence of {Y}, {VAR1} must happen before {VAR2} can happen

After the first occurrence of {X}, {VAR1} must happen before {VAR2} can happen

Between the occurrence of {X} and {Y}, {VAR1} must happen before {VAR2} can happen

Response

Across all time, {VAR2} must happen sometime after each occurrence of {VAR1}

Before the first occurrence of {Y}, {VAR2} must happen sometime after each occurrence of {VAR1}

After the first occurrence of {X}, {VAR2} must happen sometime after each occurrence of {VAR1}

Between the occurrence of {X} and {Y}, {VAR2} must happen sometime after each occurrence of {VAR1}

Duration

Minimum Duration

Across all time, whenever {VAR} happens it holds for at least {C}

Before the first occurrence of {Y}, whenever {VAR} happens it holds for at least {C}

After the first occurrence of {X}, whenever {VAR} happens it holds for at least {C}

Maximum Duration

Across all time, whenever {VAR} happens it holds for less than {C}

Before the first occurrence of {Y}, whenever {VAR} happens it holds for less than {C}

After the first occurrence of {X}, whenever {VAR} happens it holds for less than {C}

Periodic

Bounded Recurrence

Across all time, {VAR} will happen at least every {C}

Before the first occurrence of {Y}, {VAR} will happen at least every {C}

After the first occurrence of {X}, {VAR} will happen at least every {C}

Real-time Order

Bounded Response

Across all time, if {VAR1} happens then {VAR2} happens after at most {C}

Before the first occurrence of {Y}, if {VAR1} happens then {VAR2} happens after at most {C}

After the first occurrence of {X}, if {VAR1} happens then {VAR2} happens after at most {C}

Bounded Invariance

Across all time, if {VAR1} happens then it is followed by {VAR2}, which holds for at least {C}

Before the first occurrence of {Y}, if {VAR1} happens then it is followed by {VAR2}, which holds for at least {C}

After the first occurrence of {X}, if {VAR1} happens then it is followed by {VAR2}, which holds for at least {C}

9.0 Appendix B
Questionnaire

User Evaluation of an Interactive Computer System
(For each of the following questions, circle 0-9 or leave blank if question is not applicable)

OVERALL REACTIONS TO THE PROGRAM (Skip question if not applicable)

---

1) Terrible Wonderful
1 2 3 4 5 6 7 8 9 10

2) Frustrating Satisfying
1 2 3 4 5 6 7 8 9 10

3) Dull Stimulating
1 2 3 4 5 6 7 8 9 10

4) Difficult Easy
1 2 3 4 5 6 7 8 9 10

5) Inadequate Power Adequate Power
1 2 3 4 5 6 7 8 9 10

6) Rigid Flexible
1 2 3 4 5 6 7 8 9 10

REACTIONS TO THE SCREEN

The next set of questions relates to the information presented to you and how it was presented

7) Characters on the computer screen:
Hard to read Easy to read
1 2 3 4 5 6 7 8 9 10
Comments:

8) Highlighting on the screen simplifies task:
Not at all Very much
1 2 3 4 5 6 7 8 9 10

Comments:

9) Organization of information on screen:

a) Confusing Very clear

1 2 3 4 5 6 7 8 9 10

b) Inefficient Efficient

1 2 3 4 5 6 7 8 9 10

Comments:

10) Sequence of screens:

Confusing Very clear

1 2 3 4 5 6 7 8 9 10

Comments:

TERMINOLOGY AND SYSTEM INFORMATION

The next set of questions relates to the terminology you encountered while using the system and the way the system conveyed information to you

11) Use of terms throughout system:

Inconsistent Consistent

1 2 3 4 5 6 7 8 9 10

Comments:

12) Computer terminology is related to the task you are doing:

Never Always

1 2 3 4 5 6 7 8 9 10

Comments:

13) Position of messages on screen:

Inconsistent Consistent

1 2 3 4 5 6 7 8 9 10

Comments:

14) Messages on screen which prompt user for input:

Confusing									Clear
1	2	3	4	5	6	7	8	9	10

Comments:

15) Computer keeps you informed about what it is doing:

Never									Always
1	2	3	4	5	6	7	8	9	10

Comments:

16) Error messages:

Unhelpful									Helpful
1	2	3	4	5	6	7	8	9	10

Comments:

LEARNING

The next set of questions relates to your experience in learning to use the system, which is a combination of the training you received and the task you performed during this assessment.

17) Learning to operate the system:

Difficult									Easy
1	2	3	4	5	6	7	8	9	10

Comments:

18) Exploring new features by trial and error:

Difficult									Easy
1	2	3	4	5	6	7	8	9	10

Comments:

19) Remembering names and use of commands:

Difficult									Easy
1	2	3	4	5	6	7	8	9	10

Comments:

20) Tasks can be performed in a straight-forward manner:

Never									Always
1	2	3	4	5	6	7	8	9	10

Comments:

21) Help messages on the screen:

Unhelpful									Helpful
1	2	3	4	5	6	7	8	9	10

Comments:

22) Supplemental reference materials:

Confusing									Clear
1	2	3	4	5	6	7	8	9	10

Comments:

SYSTEM CAPABILITIES

The next set of questions relates to the capabilities of the system based on your use during the assessment

23) System speed:

Too slow									Fast enough
1	2	3	4	5	6	7	8	9	10

Comments:

24) System reliability:

Unreliable									Reliable
1	2	3	4	5	6	7	8	9	10

Comments:

25) System should be used:

Infrequently Frequently

1 2 3 4 5 6 7 8 9 10

Comments:

26) Correcting your mistakes:

Difficult Easy

1 2 3 4 5 6 7 8 9 10

Comments:

27) Experienced and inexperienced users' needs are taken into consideration:

Never Always

1 2 3 4 5 6 7 8 9 10

Comments:

SEQUENCE OF OPERATIONS

The next set of questions relates to the sequencing of steps you performed to create patterns and submit them to the model checker.

The sequence generally was: 1) select pattern to use, 2) select variable to define, 3) define variable by selecting from tree, 4) repeat steps 2-3 until all variables are defined, 5) submit the pattern, 6) repeat steps 1-5 until all patterns are created, 7) submit the completed patterns

28) Satisfaction with the sequence of steps:

Dissatisfied Satisfied

1 2 3 4 5 6 7 8 9 10

Comments:

29) The sequence of operations was:

Confusing Clear

1 2 3 4 5 6 7 8 9 10

Comments:

30) While performing this sequence I made:

Many Mistakes No Mistakes

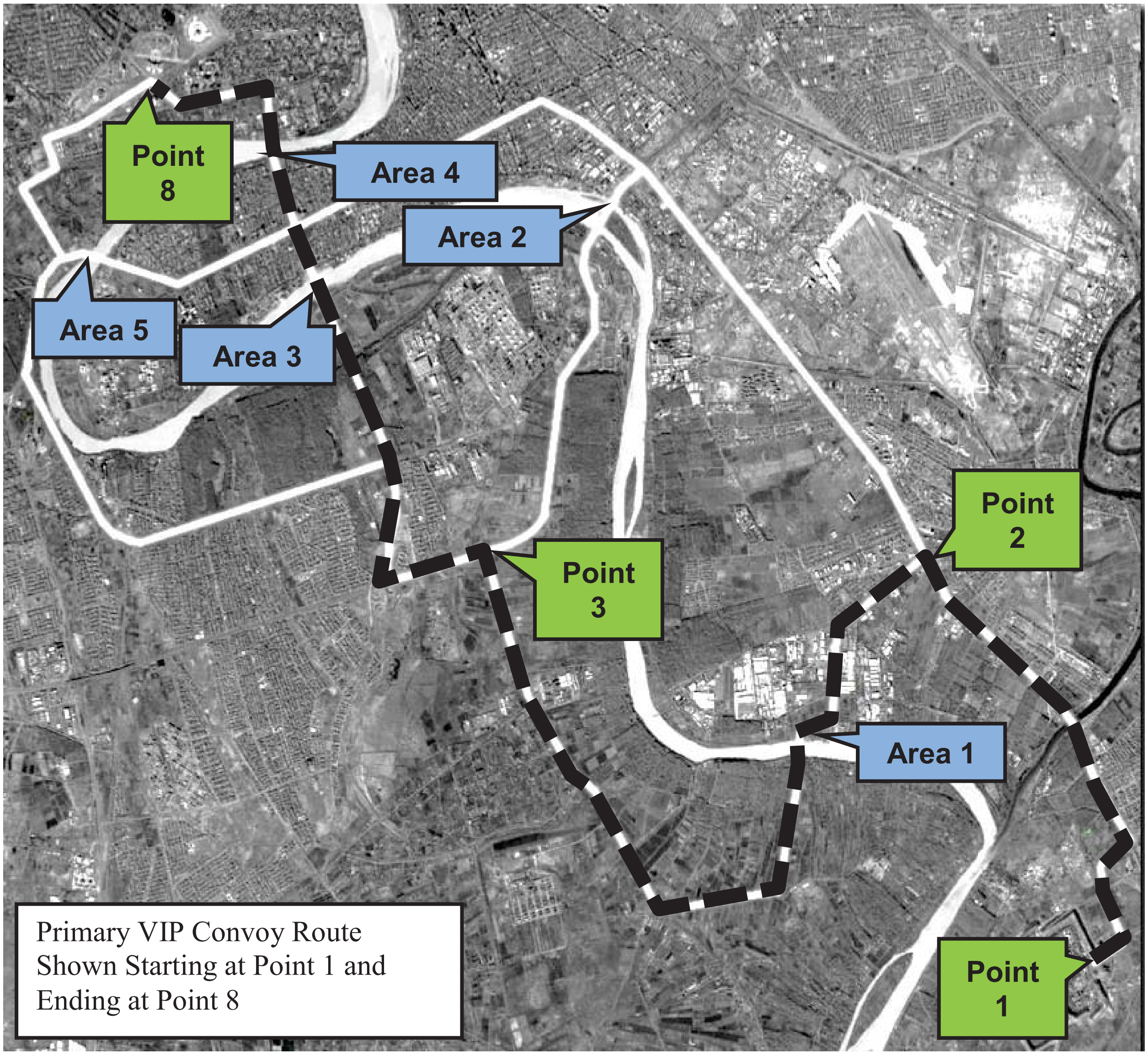
1 2 3 4 5 6 7 8 9 10

Comments:

10.0 Appendix C
Interview Questions

1. What was your workflow strategy for completing the task: build all routes first, build all patterns first, or interleave the routes and patterns?
    a. Did the design suit your strategy? How did/didn't it?
    b. Did the design influence/shape your strategy?
2. When things were flagged by the model checker, was it clear what was wrong and how to fix it?
3. Did you find the patterns in the library easy to understand?
    a. Do you have any wording suggestions for the patterns?
4. Do you have any design ideas for organizing the pattern library (left)?
5. Do you have any design ideas for integrating the SPEC and the TSD?
6. Do you have any design ideas for the pattern editor panel (middle)?
7. What tasks/mission do you think this capability would be really helpful for?
8. Are there any other types of patterns that would be useful? (goals, objectives, or constraints you would like to convey to the automation)

11.0 Appendix D
Mission Scenario A



1) A UAV observes Point 2 prior to Ground Convoy arrival
2) A UAV will have continuous observation on Point 8 after the Convoy passes Point 3
3) UAVs in Area 4 remain for at least 5 minutes (may receive tasking from ground commander)
4) A UAV will take snapshot of bridge in Area 3 every 3 minutes after the ground convoy crosses bridge in Area 1
5) All UAVs will avoid all Restricted Operating Zones (ROZs, not shown on map)
6) BAT-22 will avoid BAT-23's altitude level
7) A UAV will provide overwatch in Area 4 while the Ground Convoy is in Area 4
8) BAT-22 arrives at Point 8 and commences overwatch within 5 minutes of Ground Convoy reaching point 8

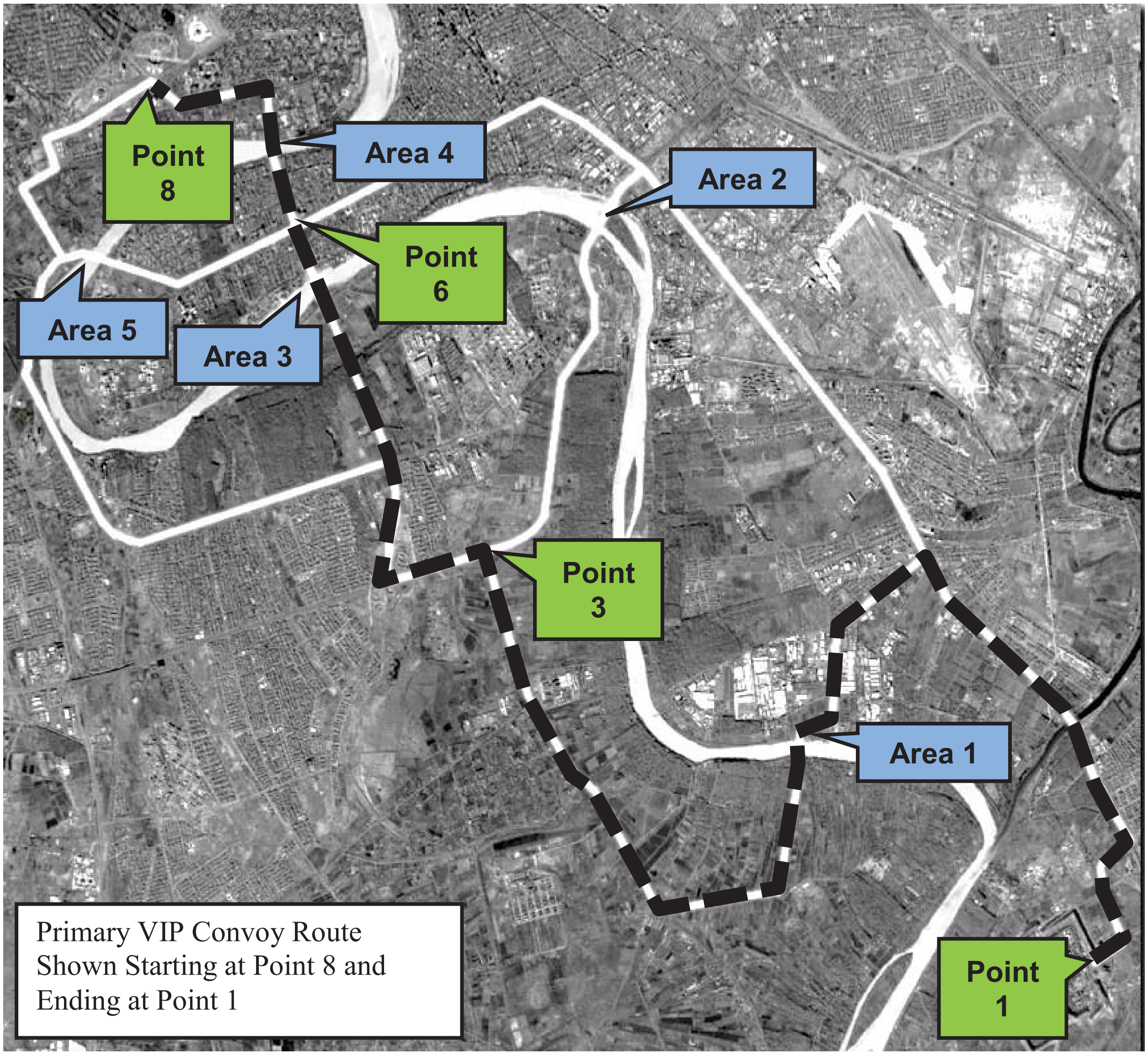
Altitude Level	Mean Sea Level (MSL)	Sensor Range
Level 1	1999 ft and Below	**Yes**
Level 2	2000 ft to 2199 ft	**Yes**
Level 3	2200 ft to 2399 ft	**Yes**
Level 4	2400 ft to 2599 ft	No
Level 5	2600 ft to 2799 ft	No
Level 6	2800 ft to 2999 ft	No
Level 7	3000 ft to 3199 ft	No
Level 8	3200 ft to 3399 ft	No
Level 9	3400 ft to 3599 ft	No
Level 10	3600 ft and Above	No

Table of Altitude Levels and Sensor Range

Location	Expected Timing (mins)	Leg Length (mins)
Point 2	10:25	10:25
Area 1	14:00	03:35
Point 9	15:10	00:50
Area 10	22:55	07:45
Point 3	28:46	05:51
Point 4	33:51	05:05
Area 3	38:12	04:21
Point 6	39:06	00:54
Area 4	41:07	02:01
Point 8	44:55	03:48

Table of Timing for VIP Path

12.0 Appendix E
Mission Scenario B



1) A UAV observes Point 6 before Ground Convoy arrival
2) UAV observes bridge in Area 3 before Ground Convoy arrival
3) The observation of bridge in Area 3 must be for at least 3 consecutive minutes
4) UAVs can enter ROZ 1 for less than 2 minutes but avoid all other ROZs
5) One UAV must observe bridge in Area 2 prior to Ground Convoy arrival at Point 3
6) A UAV is in Area 1 providing overwatch to Ground Convoy when Ground Convoy is in Area 1
7) BAT-22's altitude level should always be less than BAT-23's altitude level
8) BAT-22 arrives at Point 1 and commences overwatch within 7 minutes of Ground Convoy reaching Point 1

Altitude Level	Mean Sea Level (MSL)	Sensor Range
Level 1	1999 ft and Below	**Yes**
Level 2	2000 ft to 2199 ft	**Yes**
Level 3	2200 ft to 2399 ft	**Yes**
Level 4	2400 ft to 2599 ft	No
Level 5	2600 ft to 2799 ft	No
Level 6	2800 ft to 2999 ft	No
Level 7	3000 ft to 3199 ft	No
Level 8	3200 ft to 3399 ft	No
Level 9	3400 ft to 3599 ft	No
Level 10	3600 ft and Above	No

Table of Altitude Levels and Sensor Range

Location	Expected Timing (mins)	Leg Length (mins)
Area 4	3:36	3:36
Point 6	5:05	1:29
Area 3	6:42	1:37
Point 4	10:26	5:21
Point 3	15:49	5:23
Point 10	21:40	5:51
Point 9	29:36	7:56
Area 1	30:45	1:09
Point 2	34:19	3:34
Point 1	43:52	9:33

Table of Timing for VIP Path

13.0 LIST OF ACRONYMS

(ANOVA)	Analysis of Variance
(CI)	Confidence Interval
(CTL)	Computation Tree Logic
(CTA)	Cognitive Task Analysis
(IQR)	Inner Quartile Range
(LTL)	Linear Temporal Logic
(MSL)	Mean Sea Level
(NASA-TLX)	National Aeronautics and Space Administration Task Load Index
(NuSMV)	New Symbolic Model Verifier
(QUIS)	Questionnaire for User Interface Satisfaction
(QUIS-a)	Questionnaire for User Interface Satisfaction adapted
(ROZ)	Restricted Operating Zone
(RTCTL)	Real Time Computation Tree Logic
(SD)	Standard Deviation
(SP)	Surveillance Point
(SPEC)	Specification Pattern Editor and Checker
(SPIDER)	Specification Pattern Instantiation and Derivation EnviRonment
(TSD)	Tactical Situation Display
(UAV)	Unmanned Aerial Vehicle
(VIP)	Very Important Person
(VSCS)	Vigilant Spirit Control Station
(VTASC)	Verifiable Task Assignment and Scheduling Controller
(WPAFB)	Wright Patterson Air Force Base